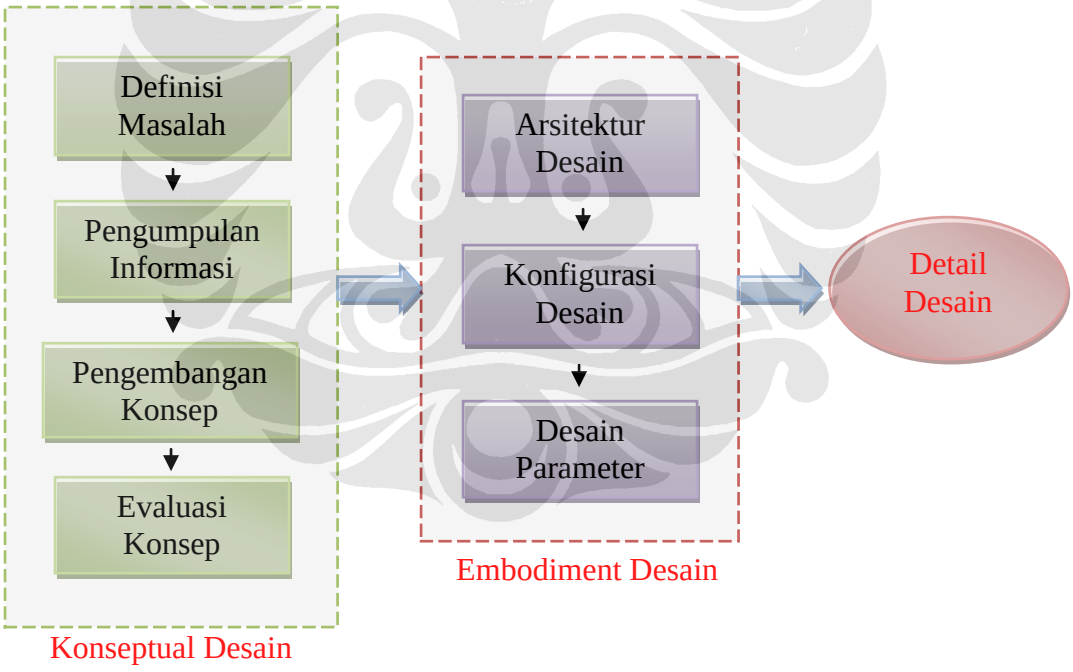
BAB III
PERANCANGAN SISTEM KONTROL
TEST BED AUTOMATIC CRUISE CONTROL

III.1 Alur Perancangan

Perancangan sistem kontrol *test bed* sistem *Automatic Cruise Control* dilakukan dengan menggunakan alur perancangan yang sistematis. Hal ini dilakukan untuk menjamin bahwa desain yang dibuat akan memiliki spesifikasi khusus yang berkaitan dengan kemudahan dalam proses manufaktur dan memiliki performa yang baik. Secara umum alur langkah perancangan terbagi kedalam tiga fase utama, yaitu konseptual desain, pembentukan desain (*embodiment*), dan detail desain.



Gambar III. 1 : Alur Perancangan [5]

Ketiga fase di atas merupakan dasar dari proses desain, yang juga merupakan permulaan dari keseluruhan proses desain itu sendiri. Fase awal ini akan membawa desain dari kemungkinan menjadi kepastian praktek di dunia nyata. Bagaimanapun juga proses desain tidak akan berhenti dengan selesainya

spesifikasi dan gambar detail *engineering*. Masih banyak keputusan-keputusan, baik teknis maupun bisnis, yang harus diambil. Termasuk bagaimana desain dimanufaktur, dipasarkan, dipelihara selama pemakaian, dan dibuang setelah produk tidak lagi digunakan.

III.2 Konseptual Desain

III.2.1 Definisi Masalah

Sistem *Automatic Cruise Control* yang dibuat merupakan sistem sederhana yang dapat dipasang sebagai fitur tambahan pada sebuah mobil (*add-on*). Oleh karena itu, desain yang berusaha dikembangkan adalah desain yang sederhana dan fleksibel sehingga mudah dalam memasang dan dapat dipasang pada berbagai jenis mobil yang ada. Faktor lain yang harus diperhatikan adalah perangkat keselamatan yang harus dapat dipenuhi oleh sistem ini. Perangkat keamanan ini amat vital mengingat sistem *Automatic Cruise Control* ini berkaitan langsung dengan keselamatan manusia dalam berkendara.

Perangkat keselamatan standar yang harus bisa diakomodir dalam sistem ini antara lain suatu alarm yang akan berbunyi dalam selang waktu tertentu. Alarm ini, selain berfungsi untuk menjaga konsentrasi pengemudi, juga berfungsi mengingatkan pengemudi untuk mengaktifkan kembali sistem tersebut. Selain itu, sistem harus memberikan respon cepat untuk berhenti bekerja saat pedal rem atau kopling (*clutch*) ditekan. Hal ini berfungsi untuk menghindari kendaraan tetap melaju tanpa kendali walaupun pedal rem telah ditekan.

Berdasarkan penjelasan di atas, dapat disimpulkan bahwa dalam sebuah sistem Automitic Cruise Control terdapat parameter input dan output yang harus dipenuhi. Parameter input yang harus dipenuhi antara lain,

- Tombol aktifasi, yang berfungsi untuk mengaktifkan sistem ini
- Keadaan dari pedal gas dan pedal kopling (*clutch*).

Sedangkan parameter outputnya terdiri dari,

- Alarm yang berbunyi dalam selang waktu tertentu.

- Sistem bekerja saat tombol aktifasi ditekan. Selain itu, pada saat alarm berbunyi, penekanan tombol aktifasi selain untuk mengaktifkan kembali sistem juga untuk menghentikan bunyi dari alarm tersebut.
- Respon untuk menghentikan sistem saat pedal gas atau kopling (*clutch*) ditekan.

Untuk mengatur hal itu semua, dibutuhkan suatu mekanisme kontrol yang berfungsi untuk mengatur masukan-masukan yang berupa kondisi tertentu yang telah diperhitungkan, yang pada akhirnya memicu aksi-aksi dari sistem *Automatic Cruise Control* ini. Sistem kontrol yang akan dibuat merupakan sistem kontrol yang berbasis mikrokontroler. Pemilihan sistem kontrol dengan menggunakan mikrokontroler didasarkan pada kemudahan dalam proses pembuatan sehingga tidak memerlukan biaya yang besar (*low cost controller*).

III.2.2 Konsep Sistem Kontrol

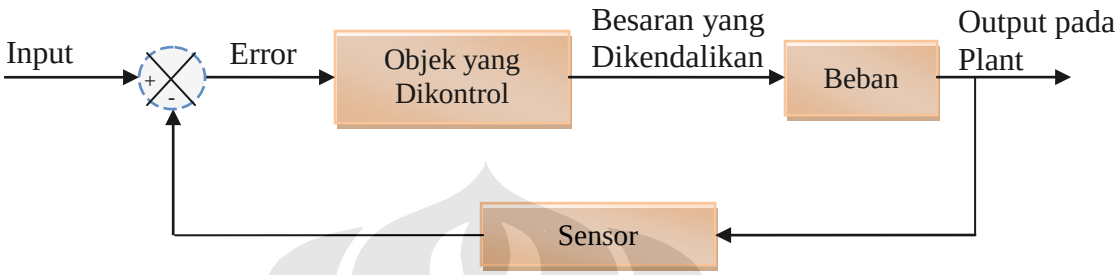
Sistem kontrol merupakan dasar utama dalam sebuah sistem mekatronika. Ada beberapa jenis sistem kontrol yang biasa dipakai sesuai dengan fungsi dan tujuan tertentu. Salah satunya adalah sistem kontrol *open loop* yang diartikan sebagai sistem kontrol terbuka, yaitu hasil output dari kontroler yang hanya bergantung pada input kontroler. Hubungan antara input dan output pada sistem kontrol *open loop* ini dapat dijelaskan dalam diagram kontrol berikut :

Gambar III. 2 : Diagram Blok Open-Loop Kontroler [7]

Input dapat didefiniskan sebagai nilai referensi yang akan diolah oleh mikrokontroler untuk menghasilkan output yang selanjutnya diaplikasikan pada mekanisme proses yang dituju.

Sedangkan sistem kontrol yang kedua adalah sistem kontrol tertutup (*close loop controller*), yang memiliki karakteristik yang berbeda dari sistem

kontrol sebelumnya. Pada sistem kontrol *close loop*, hasil output yang dihasilkan oleh kontroler akan dijadikan parameter koreksi bagi kontroler untuk mengeksekusi output selanjutnya. Dapat dikatakan pula sistem kontrol ini adalah sistem kontrol dengan umpan balik (*feedback*) seperti yang digambarkan pada ilustrasi berikut :



Gambar III. 3 : Diagram Blok Close-Loop Kontroler [7]

Sesuai dengan parameter input dan ouput yang harus diakomodir, sistem kontrol yang akan digunakan dalam sistem *Automatic Cruise Control* ini adalah sistem kontrol *close-loop*. Pengendaliannya cukup sederhana karena hanya menggunakan prinsip ON/OFF. Misalnya, pada saat sistem hidu, akan selalu dievaluasi apakah terjadi penekanan pada pedal rem atau kopling (*clutch*). Apabila sensor pada pedal rem atau kopling (*clutch*) dalam keadaan ON (salah satu pedal diinjak) maka akan mengakibatkan sistem berhenti bekerja secara otomatis.

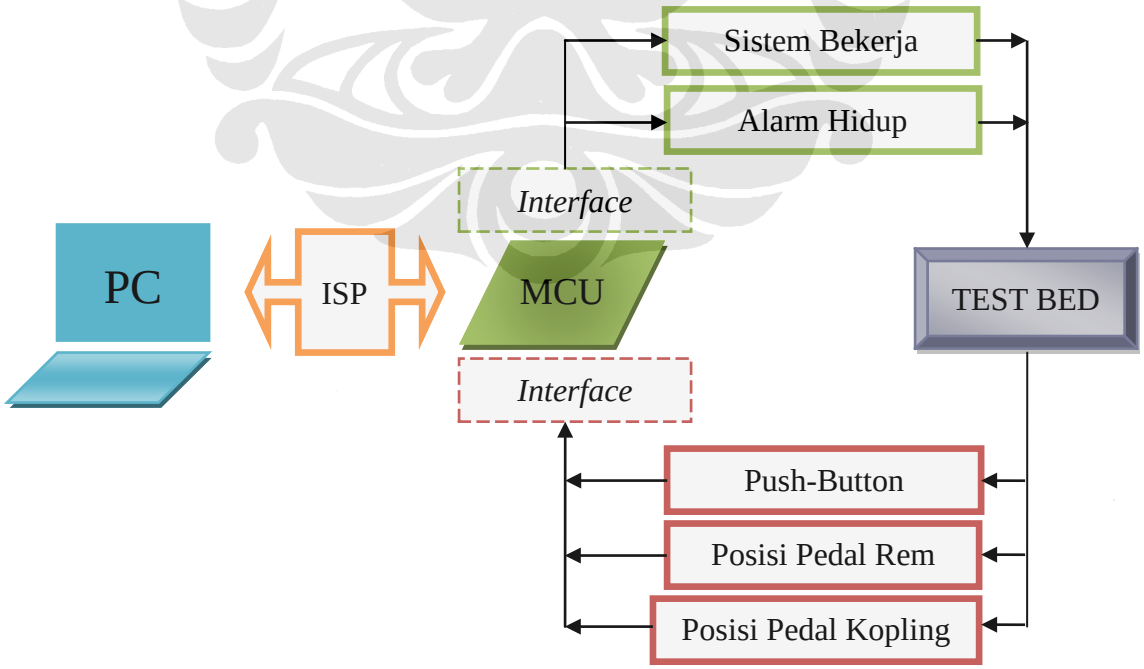
III.2.3 Sistem Elektronik

Sebuah mikrokontroler, untuk dapat bekerja dengan baik harus disuplai dengan sumber tegangan yang sesuai dengan spesifikasinya. Selain membutuhkan sumber tegangan, mikrokontroler juga membutuhkan suatu *crystal oscillator* yang berfungsi untuk mengatur “detak” dari mikrokontroler tersebut. Frekuensi dari *crystal oscillator* yang dipakai akan sangat mempengaruhi kerja dari mikrokontroler. Koneksi dengan perangkat input/output diperlukan untuk mengintegrasikan mikrokontroler dengan perangkat tersebut.

Pemilihan sensor harus dilakukan dengan baik agar tujuan yang diinginkan dengan adanya sensor tersebut dapat dipenuhi. Sensor harus memberikan respon yang baik dan akurat terhadap setiap kondisi yang dapat terjadi. Pemilihan sensor ini juga harus memperhatikan faktor harga, sensor harus dipilih sehingga dengan harga yang sehemat mungkin, fungsi yang diinginkan dapat dipenuhi.

III.3 *Embodiment* Desain

Sistem kontrol yang dibuat menganut sistem kontrol *close-loop*. Parameter input yang dijadikan *feedback* adalah posisi dari pedal rem dan pedal kopling (*clutch*). Parameter input lainnya adalah sebuah *push-button* yang digunakan untuk mengaktifkan sistem ini, baik pada saat awal ataupun saat pengaktifan kembali setelah selang waktu tertentu. Output yang ada berupa bekerjanya sistem saat *push-button* ditekan dan adanya alarm yang berbunyi selama selang waktu tertentu. Parameter-parameter tersebut sangat berkaitan dengan faktor keselamatan sehingga harus dapat bekerja dengan baik. Keseluruhan fungsi dan alur kerja dari sistem kontrol pada *test bed* tersebut dapat dilihat pada skematik berikut.

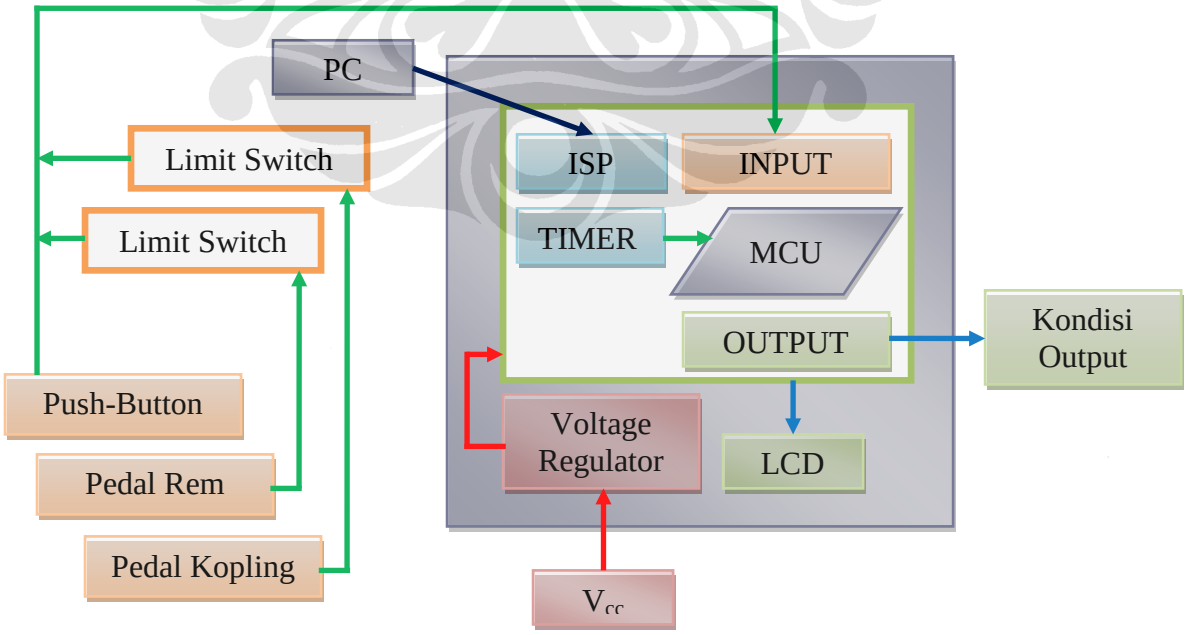


Gambar III. 4 : Diagram Skematik Sistem Kontrol

Sistem kontrol tersebut memiliki mekanisme pembacaan parameter input dan pengeluaran parameter output yang keduanya akan digunakan pada *test bed*. Otak yang mengatur sistem kontrol ini adalah *microcontroller unit* (MCU) yang memiliki fungsi antarmuka untuk menerima dan meneruskan sinyal dari sensor dan parameter output. MCU ini bisa dihubungkan ke *personal computer* (PC) untuk komunikasi dua arah. PC digunakan untuk memasukkan program ke MCU. Antarmuka yang digunakan antara PC dan MCU adalan ISP.

Pada modul *test bed* dapat dimasukkan mekanisme tampilan parameter input output tersebut. Mekanisme tersebut menggunakan *liquid crystal display* (LCD) dan *light emiting diode* (LED). LCD digunakan untuk menampilkan *mode* output apa yang sedang terjadi. LED banyak digunakan untuk memberi sinyal peringatan, misalnya saja saat alarm berbunyi dapat disertai juga dengan nyala LED berwarna merah karena warna merah banyak digunakan sebagai tanda peringatan.

Sinyal input dikirim oleh *push-button* dan sensor posisi dari pedal rem dan kopling (*clutch*). Sinyal input ini akan menentukan output yang akan dilakukan oleh sistem. Perubahan yang terjadi terhadap sensor-sensor ini akan cepat dideteksi oleh mikrokontroler sehingga dapat dilakukan eksekusi output yang sesuai dalam waktu yang cepat.



Gambar III. 5 : Modul Sistem Kontrol

Pemilihan input/output dan mikrokontroler sebagai sistem kontrol sistem ini didasarkan pada faktor kemudahan dalam pendeteksian parameter serta kemudahan *interfacing* dan pemrograman *logic* kontroler. Mikrokontroler yang digunakan adalah ATTiny 2313 merupakan mikrokontroler 8-bit jenis ATMEL AVR.

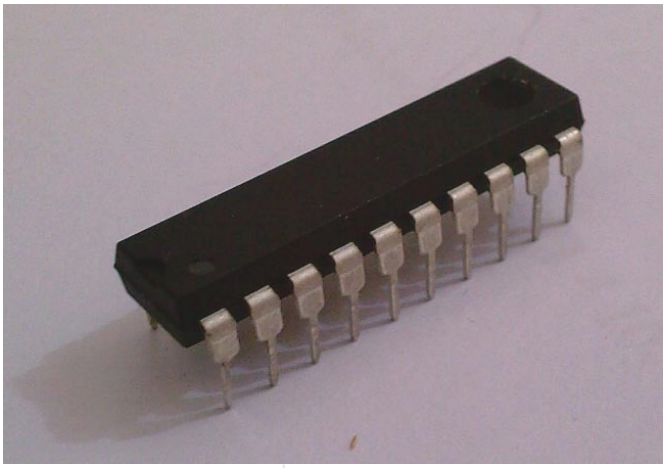
III.4 Detail Design

III.4.1 *Microcontroller Interface*

Untuk membuat mikrokontroler bekerja, perlu dibuat suatu perangkat *interface* yang menghubungkan mikrokontroler ke bagian-bagian lainnya. Bagian-bagian yang membutuhkan *interface* dengan mikrokontroler antara lain *power supply*, perangkat input, dan perangkat output.

Mikrokontroler yang dipakai adalah jenis ATTiny 2313 yang mempunyai spesifikasi sebagai berikut;

- Memori :
  - ✓ 2K *bytes* untuk *In-System Self Programmable Flash* dengan ketahanan untuk *write/erase* sampai 10.000 kali.
  - ✓ 128 *bytes* untuk *In-System Programmable EEPROM* dengan ketahanan untuk *write/erase* sampai 10.000 kali.
  - ✓ 128 *bytes* internal SRAM.
- Tegangan yang dibutuhkan saat bekerja : 2.7 - 5.5 Volt.
- Fitur *peripheral*;
  - ✓ 1 buah *Timer/Counters* 8-bit dengan *prescaler* dan *compare mode* yang terpisah.
  - ✓ 1 buah *Timer/Counters* 16-bit dengan *prescaler, compare mode*, dan *capture mode* yang terpisah.
  - ✓ 4 buah *PWM Channels*
  - ✓ *Analog Comparator*
  - ✓ *Watchdog Timer*
  - ✓ USI – *Universal Serial Interface*

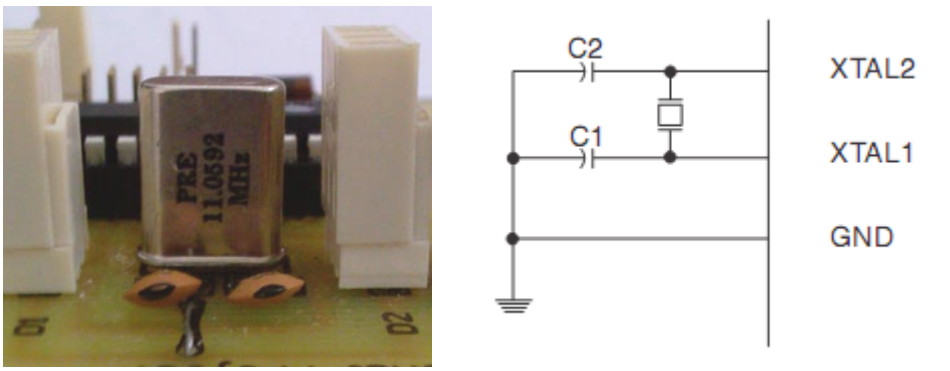


Gambar III. 6 : ATTiny 2313

Mikrokontroler membutuhkan suatu *crystal oscillator* untuk dapat menjalankan fungsinya. Frekuensi dari *crystal oscillator* ini akan sangat mempengaruhi kerja dari mikrokontroler, terutama pada perintah Timer/Counter yang berhubungan langsung dengan frekuensi tersebut. Pada desain sistem kontrol ini, *crystal oscillator* yang dipilih memiliki frekuensi 11.0592 MHz.

Tabel III. 1 : Mode Operasi dari Crystal Oscillator [3]

CKSEL3..1	Frequency Range[1] (MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals (pF)
100[2]	0.4 - 0.9	–
101	0.9 - 3.0	12 - 22
110	3.0 - 8.0	12 - 22
111	8.0 -	12 - 22



Gambar III. 7 : Pemasangan Crystal Oscillator pada Mikrokontroler [3]

Untuk mengetahui kondisi tegangan yang masuk ke mikrokontroler, antara input tegangan dan mikrokontroler dipasang sebuah LED. Tegangan yang masuk ke dalam mikrokontroler harus sesuai dengan kisaran tegangan yang disebutkan dalam spesifikasi mikrokontroler tersebut. Pengaturan sumber tegangan ini akan dibahas dalam bagian selanjutnya.

Desain *board* sistem kontrol secara lengkap dapat dilihat dalam gambar di bawah ini;

Gambar III. 8 : Skematik PCB Sistem Kontrol

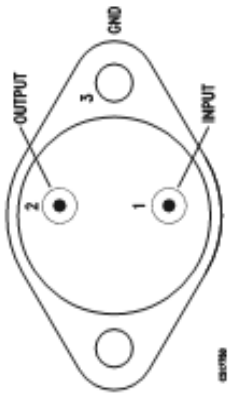
III.4.2 Pengatur Tegangan

Test bed yang dirancang memerlukan berbagai macam sumber tegangan dari masing-masing komponen seperti kontroler dan aktuator. Hal pertama yang harus diperhatikan adalah tegangan yang dibutuhkan oleh mikrokontroler. Tagangan ini tidak boleh lebih apalagi kurang karena dapat menyebabkan kerja mikrokontroler yang tidak sempurna.

III.4.2.1 Pembagi Tegangan

Sebuah pembagi tegangan memiliki tegangan keluaran yang sesuai (linear) dengan tegangan input. Besarnya perbandingan antara tegangan output dengan tegangan input disebut dengan *gain*.

Berdasarkan persamaan diatas [7], tegangan keluaran dapat dihitung dengan menggunakan rumus [7]:

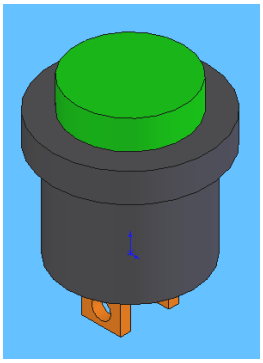


Gambar III. 9 : Skematik LM323[8]

Pada saat pemasangan di mobil, tegangan input berasal dari *accu* mobil yang mempunyai tegangan 12VDC. Keadaan ini hampir sama dengan *power supply* yang dipasang pada *test bed*. *Power supply* yang dipasang pada *test bed* juga berupa batere yang memiliki tegangan 12VDC. Setelah melalui *voltage regulator* LM323, tegangan yang keluar adalah 5 VDC yang sesuai dengan tegangan operasi dari mikrokontroler ATTiny 2313, yaitu antara 2.7 sampai 5.5 VDC. Untuk menghindari panas yang berlebih, pada LM323 biasanya dipasang *heat sink* untuk membantu pembuangan panas yang dihasilkan.

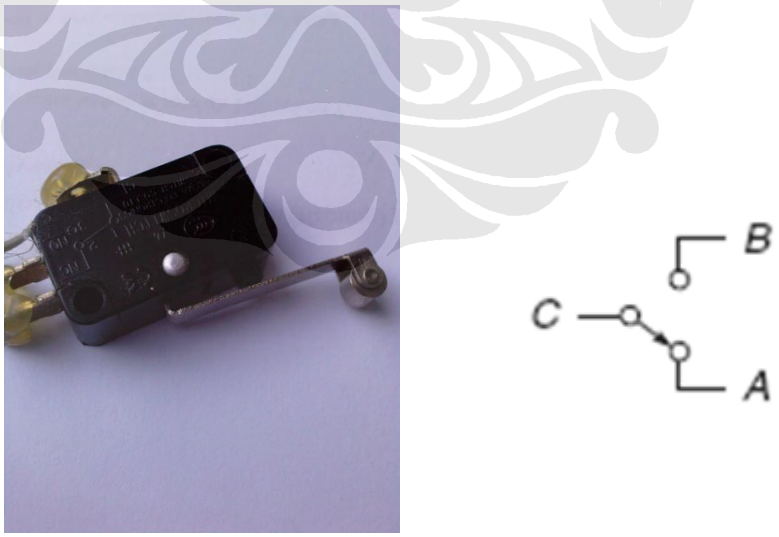
III.4.3 Komponen Input Sinyal

Ada tiga jenis input yang akan mempengaruhi kerja dari sistem secara keseluruhan. Input itu adalah *push-button*, posisi dari pedal rem, dan posisi dari pedal kopling (*clutch*). Setiap input mempunyai fungsi tersendiri. *Push-button* berfungsi untuk mengaktifkan sistem dan menghentikan bunyi alarm. Begitu *push-button* ditekan, sinyal akan ditangkap oleh mikrokontroler sehingga sistem akan bekerja. *Push-button* hanya akan mengirimkan sinyal sekali pada saat ditekan. Untuk menjaga agar sistem tetap bekerja walaupun *push-button* dilepas, dalam program yang dibuat penekanan *push-button* akan berpengaruh pada variabel tertentu.



Gambar III. 10 : Push-Button

Untuk mengetahui posisi dari pedal rem dan pedal kopling (*clutch*) cukup dengan menggunakan *limit switch*. *Limit switch* akan mengirimkan sinyal ke mikrokontroler pada saat pedal rem dan pedal kopling (*clutch*) ditekan. *Limit switch* merupakan jenis *switch* SPDT (*Single Pole Double Throw*) sehingga dapat dibuat menjadi mode *normaly open* atau *normaly close* tergantung dari penempatan kutub-kutubnya. Pada aplikasi untuk mengetahui posisi dari pedal rem dan kopling, *limit switch* dibuat bersifat *normaly close* sehingga pada saat tidak di tekan akan menghasilkan sinyal ke mikrokontroler. Hal ini untuk mempercepat respon mikrokontroler terhadap penekanan dari pedal rem dan kopling (*clutch*).



Gambar III. 11 : Limit Switch

III.4.4 Komponen Output

Output yang dijalankan oleh mikrokontroler ada 3 jenis, yaitu bekerjanya sistem (*aktuator*), bunyi alarm, dan tampilan pada LCD. Aktuator yang dipakai membutuhkan tegangan 24 VDC, sedangkan keluaran tegangan dari mikrokontroler hanya 5 VDC. Oleh karena itu, dalam *test bed* yang dibuat, sumber tegangan dari aktuator berdiri sendiri bukan berasal dari mikrokontroler. Pengaturan output oleh mikrokontroler dilakukan dengan menggunakan sebuah *relay*. *Relay* ini merupakan jenis relay SPST (*Single Pole Single Throw*) yang berfungsi sebagai saklar yang akan menghubungkan aktuator dengan sumber tegangan apabila ada keluaran dari mikrokontroler. Bentuk *relay* ini ditunjukkan pada gambar di bawah ini;

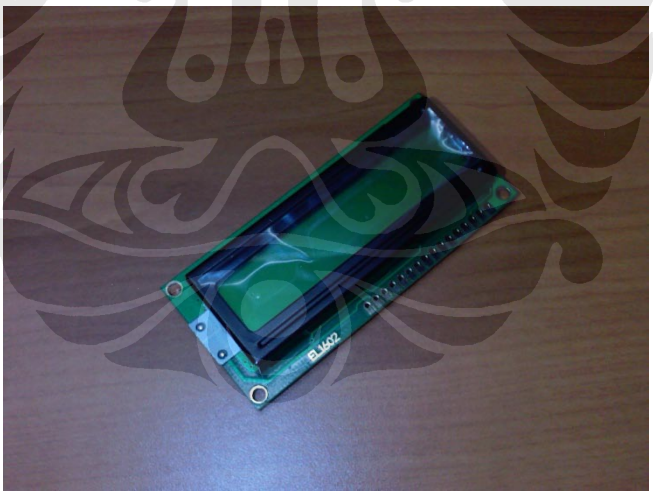
Gambar III. 12 : Relay SPST

Relay ini dapat menghubungkan tegangan sampai 24 VDC dengan tegangan pemicu 5 VDC.

Untuk menghasilkan suatu bunyi alarm digunakan sebuah *buzzer*. *Buzzer* ini bisa beroperasi pada tegangan 5 VDC seperti yang dihasilkan output mikrokontroler sehingga dapat dihubungkan langsung ke mikrokontroler. Waktu *buzzer* berbunyi dan lamanya diatur melalui fitur timer dari mikrokontroler.

Gambar III. 13 : Buzzer

Seperti halnya *buzzer*, LCD dapat langsung mendapatkan sumber tenaga dari mikrokontroler. LCD berfungsi untuk menampilkan keadaan input dan output yang sedang terjadi. Hal ini berguna untuk mengetahui kesalahan yang terjadi baik pada sinyal input ataupun pada sinyal output. Tampilan LCD masih berupa *dot matriks* dengan dua baris kumpulan karakter dimana masing-masing baris dapat menampung 16 karakter.



Gambar III. 14 : LCD

BAB IV

PEMROGRAMAN SISTEM KONTROL *TEST BED AUTOMATIC CRUISE CONTROL*

IV.1 Inisialisasi Input dan Output

Pada bagian sebelumnya telah disebutkan parameter input dan output yang harus dapat diintergrasikan ke mikrokontroler. Parameter input/output ini diintergrasikan ke mikrokontroler sehingga dihasilkan suatu siklus *close loop* sebagai sistem kontrol yang dipakai.

Parameter-parameter input yang dikontrol oleh mikrokontroler antara lain;

1. *Push button*. Input dari *push button* akan dihubungkan dengan pin D2. Input dari push-button akan mengakibatkan timer 0 mulai mengitung sekaligus menghidupkan *relay* dan mematikan *buzzer*.
2. *Limit switch* pedal rem dihubungkan dengan pin D3. Input dari *limit switch* ini akan menghentikan *relay* sehingga aktuator berhenti bekerja dan juga akan menghentikan timer 0.
3. *Limit switch* pedal kopling dihubungkan dengan pin D4. Input dari *limit switch* ini akan menghentikan *relay* sehingga aktuator berhenti bekerja dan juga akan menghentikan timer 0.

Parameter-parameter output yang dikontrol dengan mikrokontroler antara lain;

1. *Relay* untuk menghidupkan aktuator. *Relay* ini dihubungkan dengan port B3.
2. *Buzzer* yang akan berbunyi setelah selang waktu tertentu dari saat *push-button* ditekan. *Buzzer* dihubungkan dengan pin D5.
3. LCD untuk menampilkan kondisi dari input dan output sistem. LCD ini memakai seluruh port B, kecuali pin B3.

Inisialisasi dalam program untuk menggambarkan kondisi input dan output serta pin-pin yang digunakan untuk input dan output tersebut dapat dinyatakan dengan,

```
#include <tiny2313.h>

// fungsi Modul Alphanumeric LCD
#asm
   .equ __lcd_port=0x18 ;PORTB
#endasm
#include <lcd.h>

#define buzzer    PORTD.5
#define relay     PORTB.3
#define button    PIND.2
#define clutch    PIND.4
#define brake     PIND.3

void main(void)
{

// Inisialisasi Port B
// Func0=Out  Func1=Out  Func2=Out  Func3=Out  Func4=Out
   Func5=Out Func6=Out Func7=Out
// State0=0   State1=0   State2=0   State3=0   State4=0
   State5=0 State6=0 State7=0
PORTB=0x00;
DDRB=0xFF;

// Inisialisasi Port D
// Func6=In   Func5=Out   Func4=In   Func3=In   Func2=In
   Func1=In Func0=In
// State6=T   State5=0   State4=T   State3=T   State2=T
   State1=T State0=T
PORTD=0x00;
DDRD=0x20;

// Inisialisasi Modul LCD
lcd_init(16);

      while(1)
          {
           //perintah-perintah yang diinginkan
          };
}
```

IV.2 Logika Pemrograman

Setiap fitur *peripheral* yang terdapat dalam sebuah mikrokontroler mempunyai cara tersendiri dalam menginisialisasi input, menggunakan, dan mengeluarkan outputnya. Proses yang benar dalam setiap langkah penggunaan fitur *peripheral* mikrokontroler ini akan membuat *peripheral* berjalan dengan benar. Kaidah tentang cara penginisialisasian input dan penggunaan dari beberapa *peripheral* yang sering digunakan telah dijelaskan dalam bab-bab sebelumnya. Penjelasan proses inisialisasi secara detail dari setiap *peripheral* yang terdapat dalam mikrokontroler ATTiny 2313 dapat dilihat pada *data sheet* ATTiny 2313.

Logika pemrograman harus disusun secara sistematik sehingga dapat mengakomodir seluruh fungsi input dan output yang telah disebutkan sebelumnya. Selain itu, program yang sistematik juga akan memudahkan saat dilakukan pengecekan ketika terjadi sesuatu yang tidak dikehendaki. Untuk itu, pembuatan program tidak dilakukan secara menyeluruh terlebih dahulu. Pembuatan potongan-potongan program akan memberikan gambaran yang lebih sistematis tentang program secara menyeluruh.

IV.2.1 Logika Pemprograman Pengaktifan Relay Selama Selang Waktu Tertentu

Relay dihidupkan dengan suatu variabel yang nilainya di-*trigger* oleh input dari *push button* yang sekaligus akan menghidupkan timer 0. Timer 0 akan menghitung sampai selang waktu 5 menit. Setelah 5 menit, timer 0 akan mengeksekusi perintah untuk menghentikan kerja dari *relay*.

Timer 0 dinyatakan dalam mode *interrupt overflow* dengan faktor *prescaler* 64. Inisialisasi yang digunakan dalam program dinyatakan sebagai berikut;

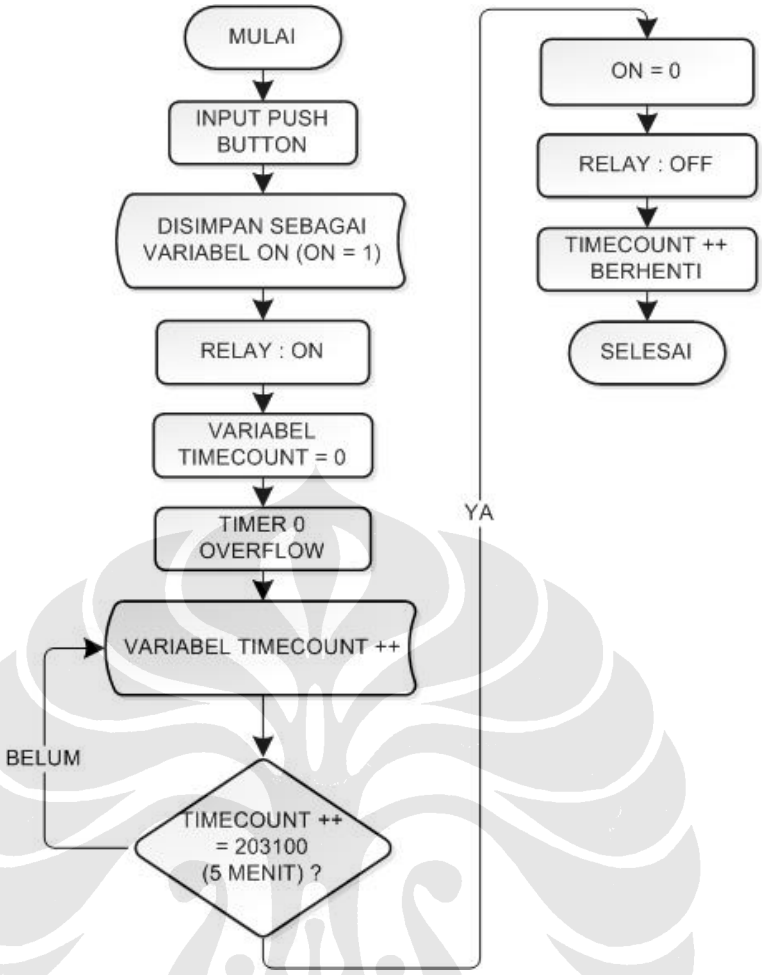
```
// Timer/Counter 0 initialization
// Clock source: System Clock
// Mode: Normal top=FFh
// OC0A output: Disconnected
// OC0B output: Disconnected
TCCR0A=0x00;
TCCR0B=0x03;
TCNT0=0x00;
```

```
OCR0A=0x00;
OCR0B=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x02;

// Global enable interrupts
#asm("sei")
```

Crystal oscillator yang digunakan mempunyai frekuensi 11.0592 MHz, dengan penggunaan prescaler 64, maka frekuensi dari Timer 0 adalah 172800 Hz. Jadi, selama 1 detik terjadi 172800 *clock* dalam mikrokontroler. Timer 0 adalah 8-bit timer, sehingga dapat menghitung sampai 255. Dengan demikian, dalam waktu 1 detik terjadi *Timer* 0 *Overflow* (*interrupt* dari *timer*) sebanyak 677 kali (172800 Hz / 255 = 677). Sehingga untuk membuat selang waktu selama 5 menit, harus terjadi *Timer* 0 *Overflow* sebanyak 203100 kali (dari 677*300 detik). Jumlah dari *Timer* 0 *Overflow* yang terjadi akan disimpan dalam suatu variabel. Logika pemrograman yang dipakai adalah;

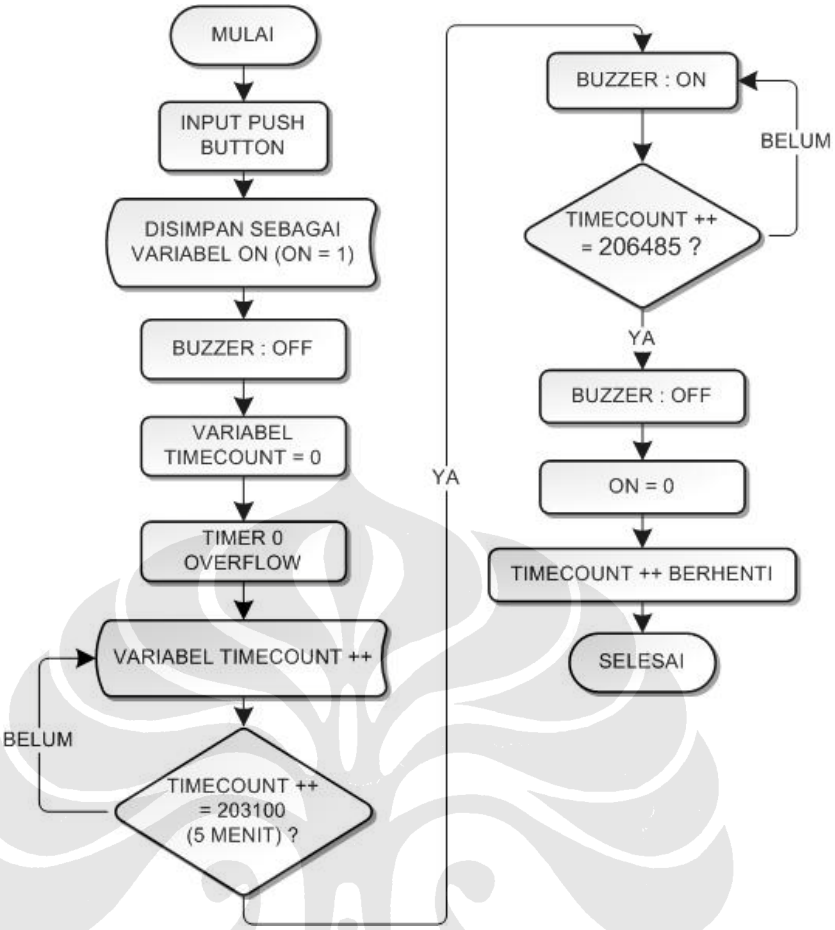


Gambar IV. 1 : Flowchart Pengaktifan Relay

IV.2.2 Logika Pemrograman Buzzer

Buzzer dinyalakan sebagai peringatan bahwa waktu aktifasi telah habis dan perlu dilakukan aktifasi lagi agar sistem tetap berjalan. *Buzzer* akan berbunyi setelah waktu 5 menit terlampaui dan hanya berbunyi selama 5 detik.

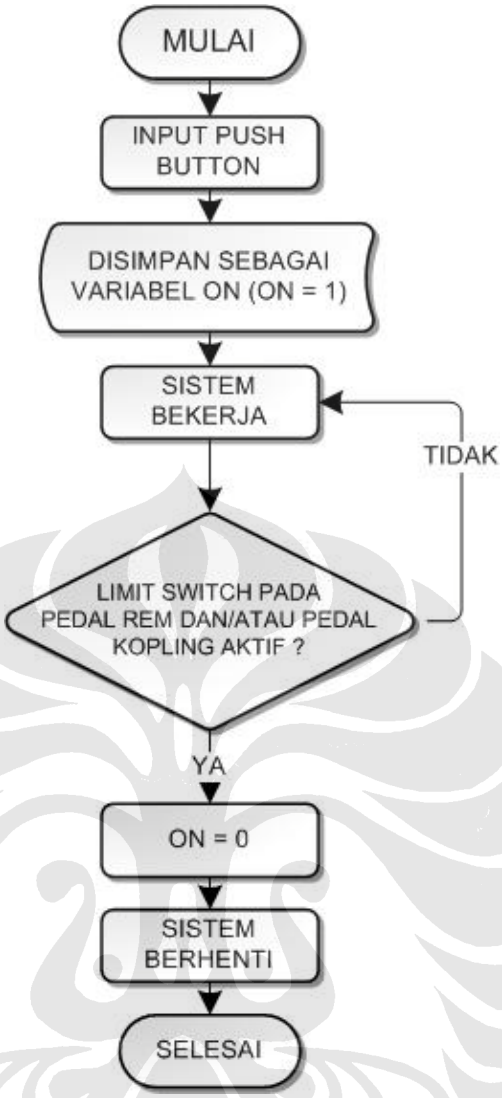
Penentuan waktu buzzer berbunyi dan lamanya dilakukan dengan fitur timer 0. Seperti disebutkan sebelumnya, untuk mencapai waktu 5 menit dibutuhkan *interrupt Timer 0 Overflow* sebanyak 203100 kali. *Buzzer* akan berbunyi saat *interrupt Timer 0 Overflow* melebihi 203100 kali dan akan berbunyi terus selama lima detik atau sampai terjadi *interrupt Timer 0 Overflow* sebanyak 206485 kali. Setelah itu, *buzzer* akan berhenti berbunyi. Penekanan *push button*, baik pada saat *buzzer* berbunyi ataupun pada saat *buzzer* tidak berbunyi akan mengembalikan proses dari awal.



Gambar IV. 2 : Flowchart Pengaktifan Buzzer

IV.2.3 Logika Pemrograman Pedal Rem dan Pedal Kopling

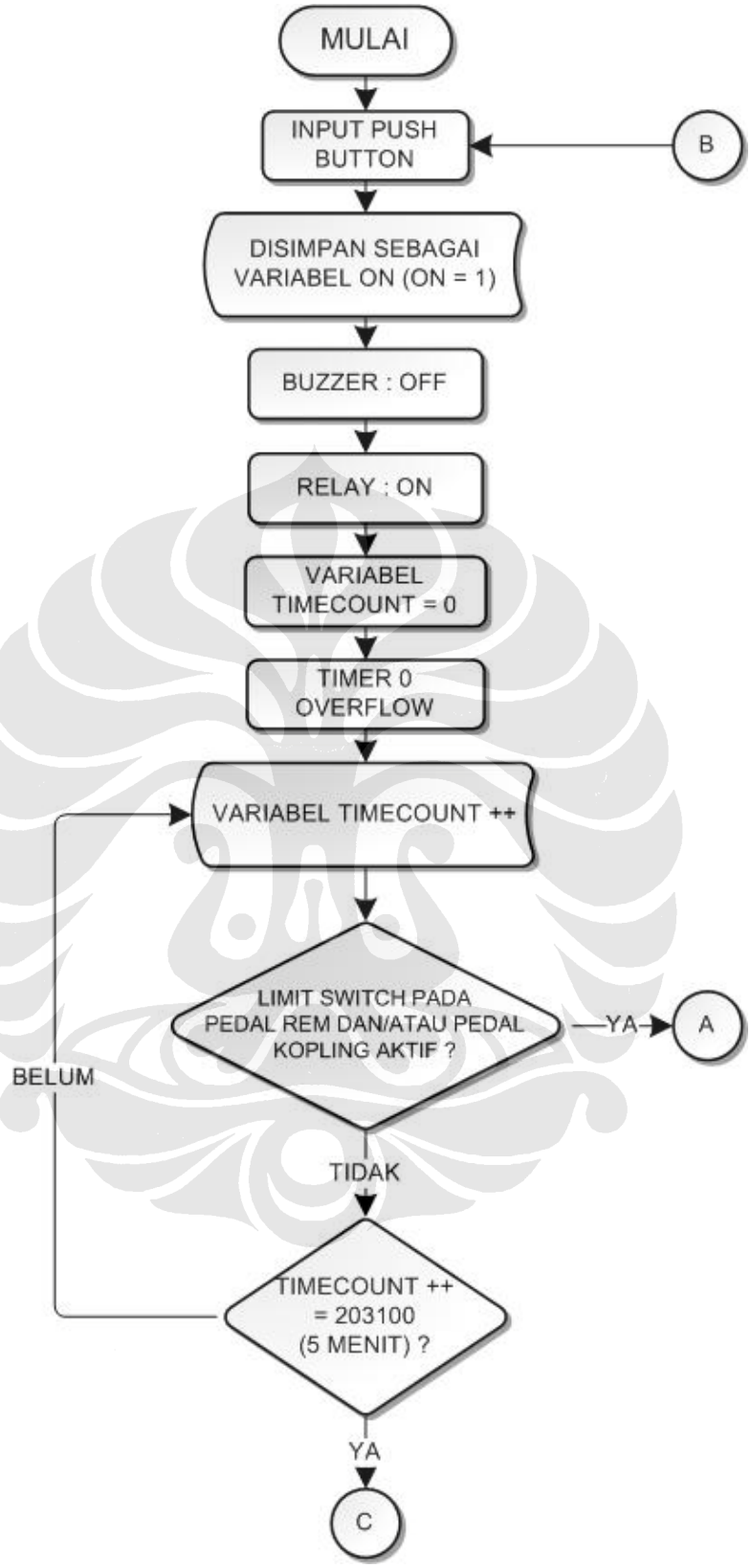
Penekanan pada pedal rem dan/atau pedal kopling akan pada saat sistem bekerja akan membuat sistem berhenti bekerja secara otomatis. Logika yang dipakai adalah;



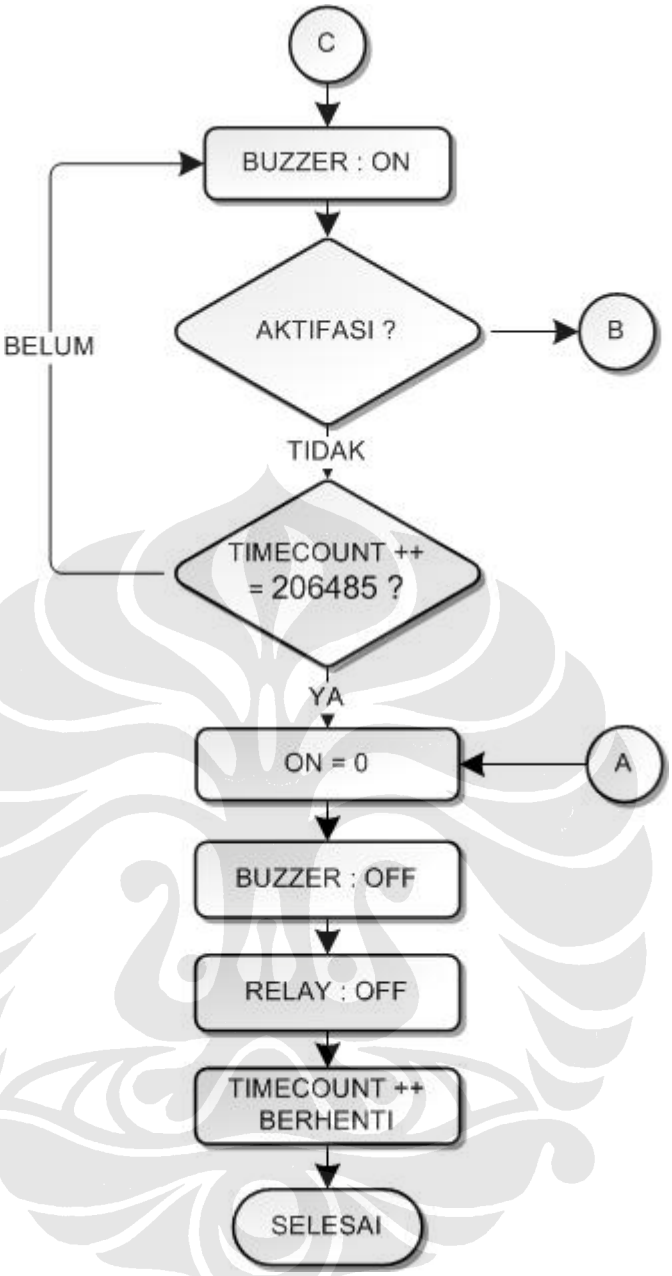
Gambar IV. 3 : Flowchart Pedal Rem dan Kopling

IV.2.4 Logika Pemrograman Keseluruhan

Dari potongan-potongan logika pemrograman yang telah ada, dapat digabungkan dan disusun sehingga membebtuk logika pemrograman yang lengkap untuk sistem kontroler *Automatic Cruise Control* yang dibuat. Logika selengkapnya adalah sebagai berikut.



Gambar IV. 4 : Flowchart Program Keseluruhan-1



Gambar IV. 5 : Flowchart Program Keseluruhan-2